Restaurant Technology Network

join. share. thrive.

RTN's Transactional Data Specification

The Benefits of Event-Driven Architecture for the Restaurant Industry

AUTHORS
Deborah Matteliano, Global Head of Restaurants, AWS
Jack Bradham, Senior Architect, AWS
Steven M. Elinson, Director AWS for Travel & Hospitality

Delighting diners is no longer a sequential operation. Whether it's fulfilling delivery orders or seating on-premises customers, restaurants are increasingly focused on earning guests' trust through all possible sales channels. The restaurant experience of the past could be reduced into a list of steps: greet customers, seat them, capture orders, check in on them, bring a bill, then turn the table over for the next guest. But in today's multi-channel restaurant, unplanned events abound, like spikes in digital orders, or viral limited time offers. Restaurants must be ready and agile in the face of these events — and so should their technology.

The old sequential approach to technology system design relied on "batch" data. This often led to missed opportunities to improve the customer experience and operational efficiency, sometimes resulting in manual processes. Batch processing, or receiving updates nightly, weekly, or monthly, is an antiquated approach when data can be fed in "real-time." Industry leaders are turning to event-driven architectures: App8, Blue Bottle Coffee, Domino's, Grubhub, KFC, and Taco Bell all find competitive advantages in these designs.

Event-driven design is simple, and all it takes is a few lines of code to publish a message and subscribe to an event:

- **Publishers (Writers) write a message (Event)**
  Examples:
  POS, time clock, inventory system, restroom occupancy sensor
- **An Event Bus (Messages) routes and presents the event as a message**
  Examples:
  New orders, reservations, employee clock-in/out
- **Subscribers (Readers) react to it**
  Examples:
  Application, partner or person needing the information, such as a loyalty program or a perpetual inventory system.

  Multiple subscribers can react to any event at the same time.

EXAMPLES:

1. New mobile order >> present the order to staff for fulfillment, move the customer to the next tier in the loyalty program, and auto-order more chicken, all at the same time, in near-real-time.
2. VIP guest sat at table # 3 >> send Manager to table to greet guest and wish them a happy birthday, send the Sommelier to the table with new Cabernet Sauvignon recommendations (the guests preference based on prior events), and notify the pastry chef that any desserts ordered for table # 3 should be sent with a birthday candle, all at the same time, in near-real-time.

Real-time event detection allows us to take immediate action. The benefits of event-driven architectures abound:

- They're flexible, systems can adapt to changing conditions.
- They're responsive, enabling real-time reactions to data.
- They are scalable, easily handling any level of events.
- And above all, they are efficient, reducing bottlenecks and driving productivity.

Just as well-coordinated restaurant staff anticipates and responds dynamically to diners' needs, businesses embracing event-driven architecture align operations with the unpredictable rhythm of real-world events. This shift enables more agile, responsive, and satisfying customer experiences, mirroring the seamless dance of a customer's journey through a well-run restaurant.

Staff

ABBY LORDEN
VP and Brand Director, *HT*
Co-Founder, RTN
973.607.1358
alorden@ensembleiq.com

ANGELA DIFFLY
Co-Founder, RTN
404.550.7789
angela@restauranttechnologynetwork.com

SANDY ANGEL
Senior Director, Technology & Information
American Hotel & Lodging Association
sangel@ahla.com

ROBERT FIRPO-CAPPIELLO
Editor in Chief, *HT*
917.208.7393
rfirpo-cappiello@ensembleiq.com

ANNA WOLFE
Senior Editor, *HT*
207.773.1154
awolfe@ensembleiq.com

KATHERINE WARE
Senior Account Executive, *HT* & RTN
785.424.7392
kware@ensembleiq.com

NOELL DIMMIG
Account Executive, *HT* & RTN
973.607.1370
ndimmig@ensembleiq.com

MOLLY MCLOONE
Brand Marketing Manager, *HT* & RTN
908.433.2796
mmcloone@ensembleiq.com

TAMMY HANSON
Membership Manager, RTN
314.570.4798
tammy@restauranttechnologynetwork.com

CHRISTOPHER BARRY
Strategic Director, Membership & Communications, EIQ
cbarry@ensembleiq.com

Table of Contents

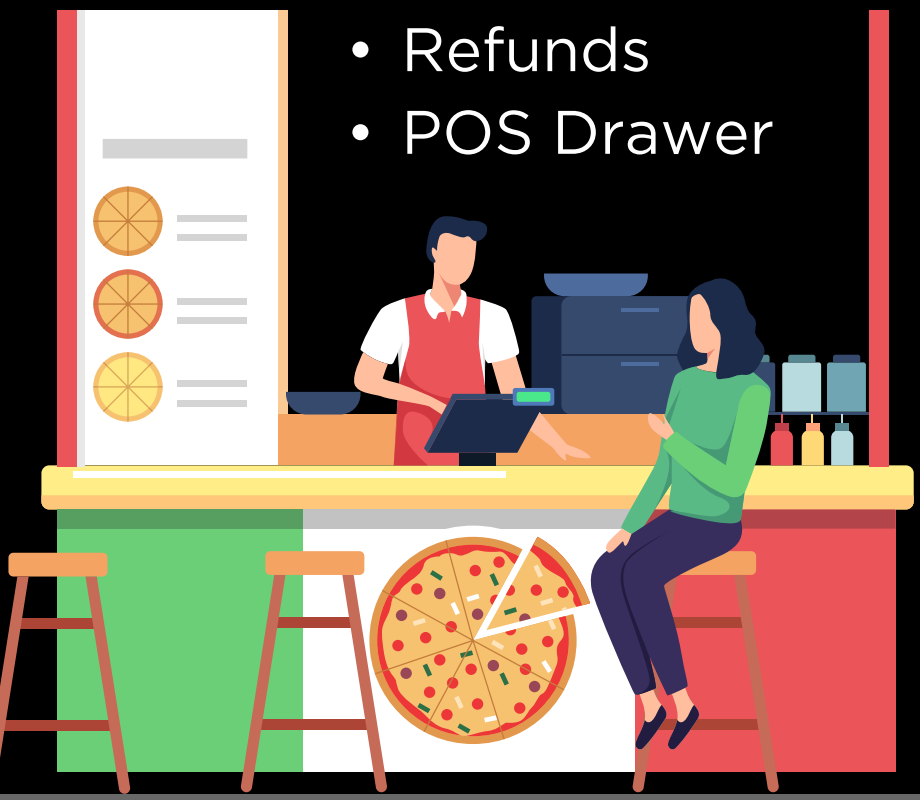
Key Contributors

RTN Mission

The Restaurant Technology Network (RTN) is a membership community solely dedicated to the restaurant technology industry. Through access to valuable benefits and powerful connections, our members shape industry standards and share technical guidance to help restaurateurs run successful businesses and better serve their customers.

Transactional Data Categories

- Ordering
- Time Card
- Reservations
- Refunds
- POS Drawer



Use Cases

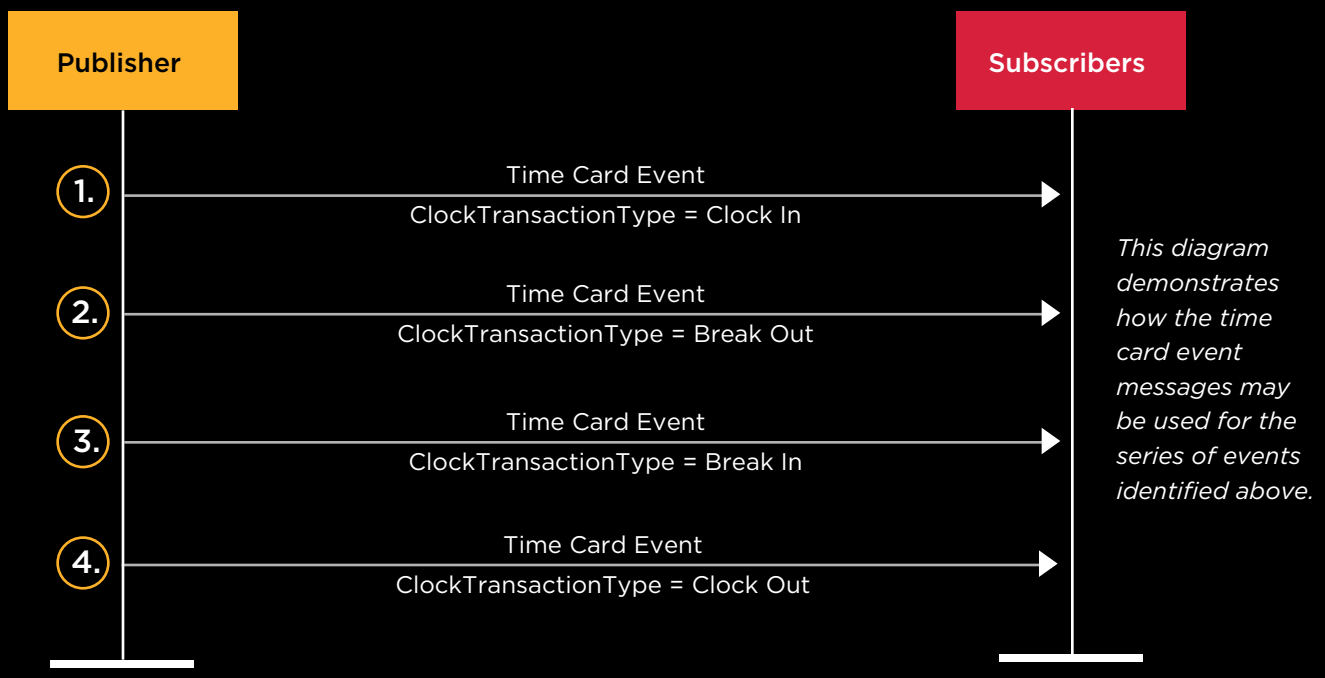
The following pages describe several end-to-end use case scenarios that utilize the transaction messages, along with diagrams for clarification.

Use Case 1

Time Card Transactions for a Single Employee

- An employee clocks in for their shift
- The employee clocks out for their break
- The employee clocks back in from their break
- The employee clocks out at the end of their shift

ASSUMPTIONS	• The employee exists in the time card system • The employee is scheduled to clock in • The payroll system is subscribed to receive updates from the time card system • The payroll system maintains a list of employee information including the employee ID
TIME CARD - CLOCK IN (1)	• The time card system publishes a Clock In time card transaction event generating a unique identifier for the new series of clock events (TimeCardCorrelationGUID) for this employee for this shift utilizing the RTN_EventNotifRQ message with the EventType = TimeCard to be received by the subscribing systems.
TIME CARD - BREAK OUT (2)	• The time card system publishes a Break Out time card transaction event re-using the unique identifier that was generated during the ClockIn event (linking the two events together via the identifier) utilizing the RTN_EventNotifRQ message with the EventType = TimeCard to be received by the subscribing systems.
TIME CARD - BREAK IN (3)	• The time card system publishes a Break In time card transaction event using the unique identifier that was generated during the Punch In event (linking the three events together via the identifier) utilizing the RTN_EventNotifRQ message with the EventType = TimeCard to be received by the subscribing systems.
TIME CARD - CLOCK OUT (4)	• The time card system publishes a Clock Out time card transaction event using the unique identifier that was generated during the Clock In event (linking the four events together via the identifier) utilizing the RTN_EventNotifRQ message with the EventType = TimeCard to be received by the subscribing systems.



This diagram demonstrates how the time card event messages may be used for the series of events identified above.

Use Case 2

Order is Created and a Cash Payment is Made (Counter Service)

- **A customer places an order through an employee at a counter.**
- **The order is sent by the ordering system to the kitchen to be made.**
- **The employee receives the cash payment from the customer, the drawer opens and the employee places the money in the till and the system closes the order.**

ASSUMPTIONS	• Menu items exist and are available in the ordering system. • Employee has clocked in and is available to take the order. • The till has previously been assigned to the employee, and is ready for transactions. • Kitchen production, labor management, and inventory management, cash management, and surveillance systems are subscribed to receive these messages.
ORDER TRANSACTION (1)	• An order is built and an orderID is created by the ordering system. • Upon completion of the order, the message is published with a status of “sent” and received by the kitchen production system utilizing the RTN_EventTransactionRQ with the EventType = Order.
ORDER TRANSACTION (2)	• The customer pays for the order. • The order is updated with payment information and is changed to a status of “closed” utilizing the RTN_EventTransactionRQ with the EventType = Order.
DRAWER TRANSACTION (3)	• The orderID is sent in the ReferencedTransaction field to link the order to the drawer transaction (payment) utilizing the RTN_EventTransactionRQ with the EventType = Drawer.

This diagram demonstrates how the order and drawer event messages may be used for the series of events as described above.

Use Case 3

Order is Created and Card Payment is Made (Online Order)

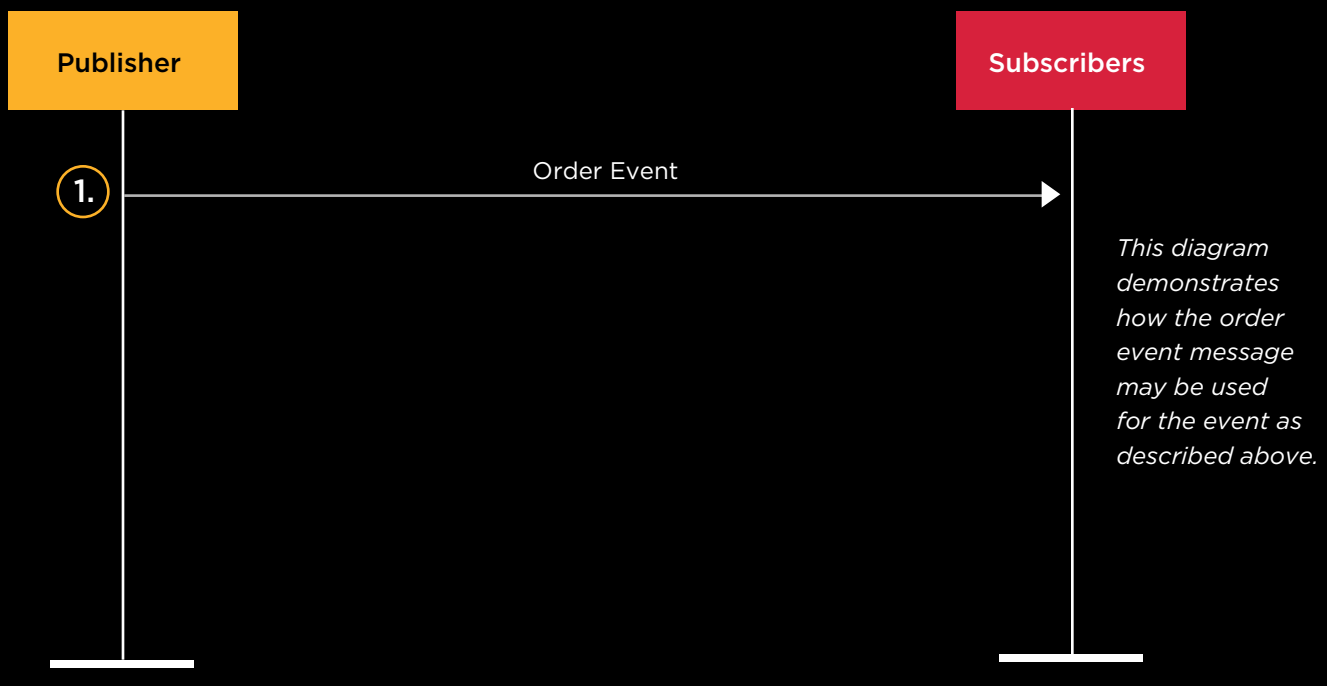
- **A customer places an order through a restaurant website including payment using a credit card.**

ASSUMPTIONS

- Menu items and pricing exist and are available in the ordering system.
- The customer has access to the online ordering system, and the ability to pay for the order using an approved method of payment.
- An order transaction engine is subscribed to receive these messages.

ORDER TRANSACTION (1)

- An order is built and an orderID is created by the ordering system.
- The customer enters credit card information for payment and finalizes the order.
- Upon completion, the order is published with a status of "closed" and received by the order transaction engine utilizing the RTN_EventTransactionRQ message with the EventType = Order.

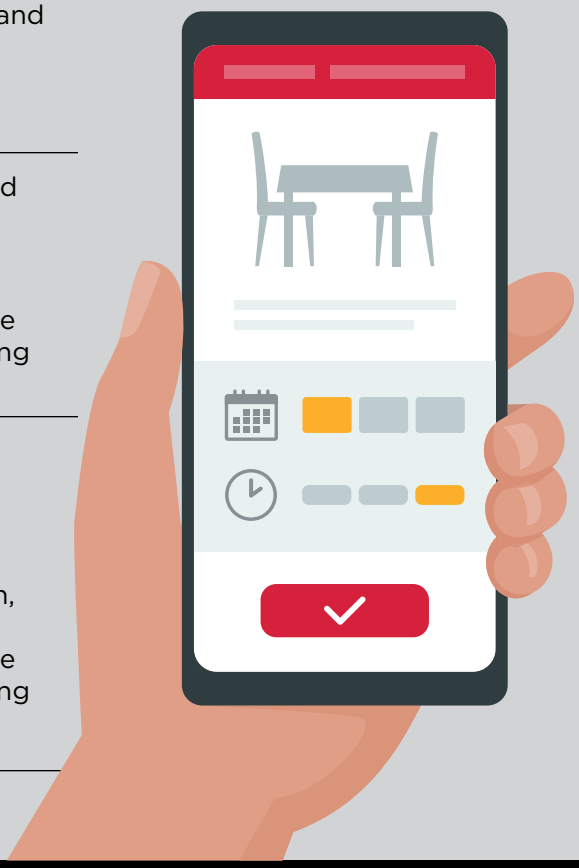


This diagram demonstrates how the order event message may be used for the event as described above.

Use Case 4 Reservations

- **A guest creates a reservation at a restaurant using a third-party online booking site for a party of 2.**
- **The guest later alters the reservation from a party of 2 to a party of 4.**

ASSUMPTIONS	• The restaurant has availability for the time and party size of the customer requests. • The restaurant is subscribed to receive the reservation messages.
RESERVATION TRANSACTION (1)	• A reservation with a reservationID is created by the online reservation system. • Upon completion of the reservation, the reservation is published utilizing the RTN_EventTransactionRQ message with the EventType = Reservation and the subscribing systems receive the message.
RESERVATION TRANSACTION (2)	• A reservation transaction with the same reservationID as the original reservation is created with the updated reservation information by the online reservation system. Upon completion of the reservation, the reservation is published utilizing the RTN_EventTransactionRQ message with the EventType = Reservation and the subscribing systems receive the message.



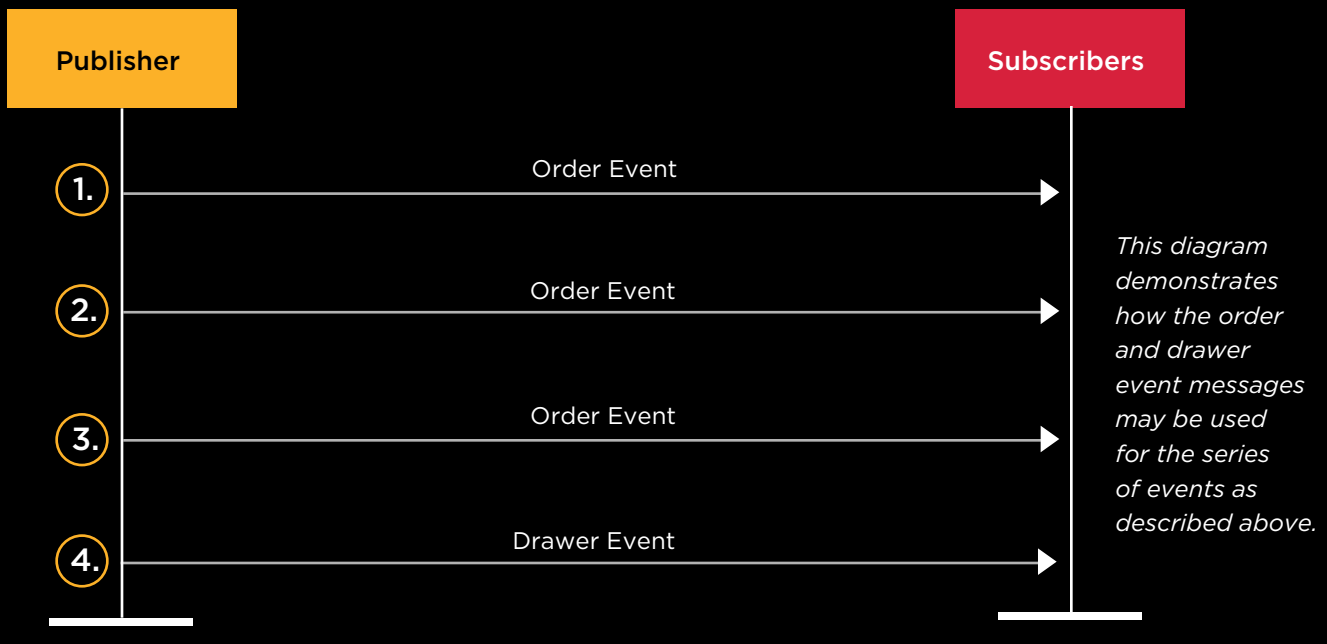
This diagram demonstrates how the reservation event message may be used for the series of events as described above.

Use Case 5

Order is Created, Item is Voided (Prior to Order Being Closed)

- **A customer places an order for a pizza and soda**
- **The customer pays for the order**
- **The drawer is opened to accept payment**
- **The customer is unhappy with the quality of the pizza and the manager refunds the price of the pizza.**
- **Cash is returned to the customer from the cash drawer.**

ASSUMPTIONS	• Menu items exist and are available in the ordering system. • Employee has clocked in and is available to take the order. • The till has previously been assigned to the employee, and is ready for transactions. • Kitchen production, labor management, and inventory management, cash management, and surveillance systems are subscribed to receive these messages.
ORDER TRANSACTION (1)	• An order is built that includes a pizza and soda and an orderID is created by the ordering system. • Upon completion of the order, the order message is published with a status of "sent" and received by the kitchen production system, utilizing the RTN_Event TransactionRQ message with the EventType = Order.
ORDER TRANSACTION (2)	• The order is modified to void the soda and add an iced tea. • Upon completion of the order, the order message is published with the status of "sent" utilizing the same order ID as the original order. • The order is received by the kitchen production system utilizing the RTN_Event TransactionRQ message with the EventType = Order.
ORDER TRANSACTION (3)	• The customer pays for the order. • The order is updated with payment information and is changed to a status of "closed" utilizing the RTN_Event TransactionRQ message with the EventType = Order.
DRAWER TRANSACTION (4)	• The orderID is sent in the ReferencedTransaction field to link the order to the drawer transaction (payment) utilizing the RTN_EventTransactionRQ message with the EventType = Drawer.

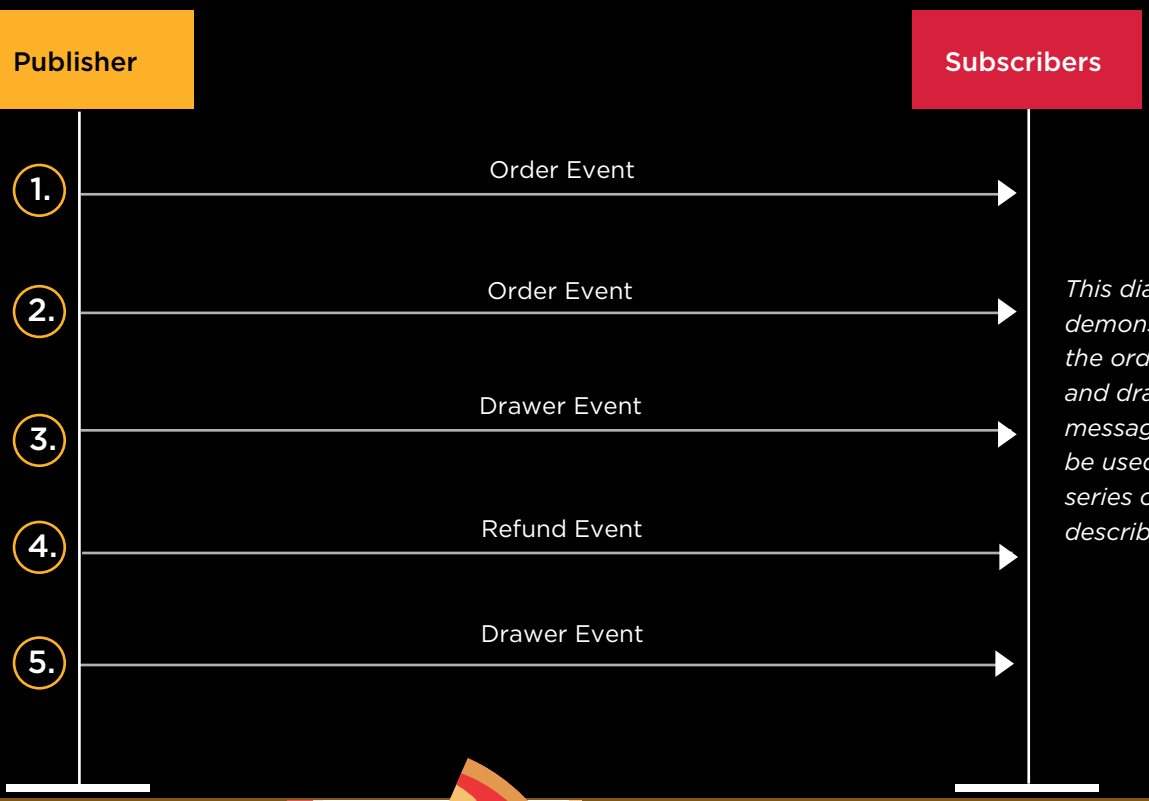


This diagram demonstrates how the order and drawer event messages may be used for the series of events as described above.

Use Case 6 Refund for Purchase on Same Business Day

- **A customer places an order for a pizza and soda**
- **The customer pays for the order**
- **The drawer is opened to accept payment**
- **The customer is unhappy with the quality of the pizza and the manager refunds the price of the pizza.**
- **Cash is returned to the customer from the cash drawer.**

ASSUMPTIONS	• Menu items exist and are available in the ordering system • Employee has clocked in and is available to take the order • The till has previously been assigned to the employee and is ready for transactions • Kitchen production, labor management, and inventory management, cash management, and surveillance systems are subscribed to receive these messages. • The refund is being made during the same business day that the order was created.
ORDER TRANSACTION (1)	• An order is built that includes a pizza and soda and an orderID is created by the ordering system. • Upon completion of the order, the message is published with a status of “sent” and received by the kitchen production system utilizing the RTN_EventTransactionRQ message with the EventType = Order.
ORDER TRANSACTION (2)	• The customer pays for the order. • The order is updated with payment information and is changed to a status of “closed” utilizing the RTN_EventTransactionRQ message with the EventType = Order.
DRAWER TRANSACTION (3)	• The orderID is sent in the ReferencedTransaction field to link the order to the drawer transaction (payment) utilizing the RTN_EventTransactionRQ message with the EventType = Drawer.
REFUND TRANSACTION (4)	• A refund is created utilizing the refund object. • The orderID of the original order is included in the OrderID field in the order object. The Payment object on the Order reflects the payment amount and information from the original order. • The Payment object on the Refund object reflects the amount of the refund as a negative amount. • The OrderItem object on the order reflects the items being refunded, in this case the pizza and the tax on the pizza reflected as negative amounts. • Upon completion of the refund, the message is published utilizing the RTN_ EventTransactionRQ message with the EvenType = Refund
DRAWER TRANSACTION (5)	• The refund is given to the customer, the orderID is sent in the ReferencedTransaction field to link the order to the drawer transaction (payment) utilizing the RTN_EventTransactionRQ message with the EventType = Drawer.

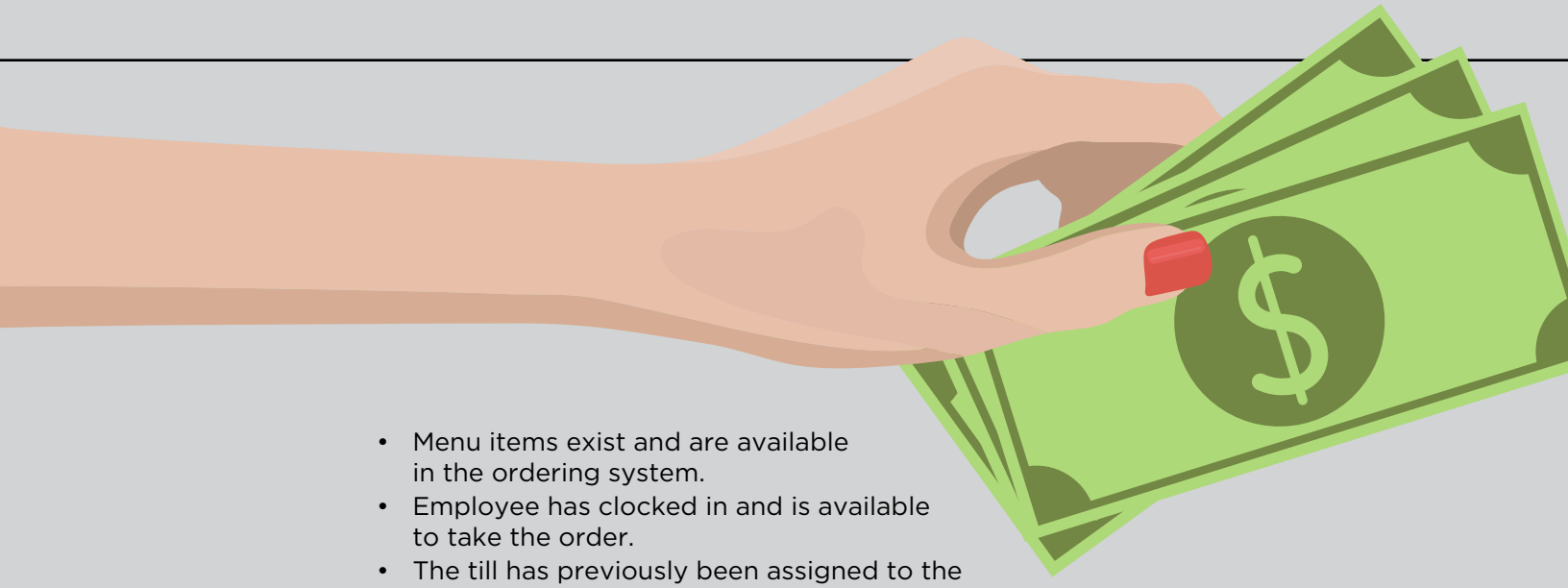


This diagram demonstrates how the order, refund and drawer event messages may be used for the series of events as described above.

Use Case 7

Refund for Purchase on a Previous Business Day

- **The customer is unhappy with the quality of the pizza they ordered the previous day and requests a refund of the price of the pizza which the store manager agrees to. The order is no longer accessible by the ordering system, so a refund is given to the customer but no relationship is made to the original order.**
- **Cash is returned to the customer from the cash drawer.**



ASSUMPTIONS

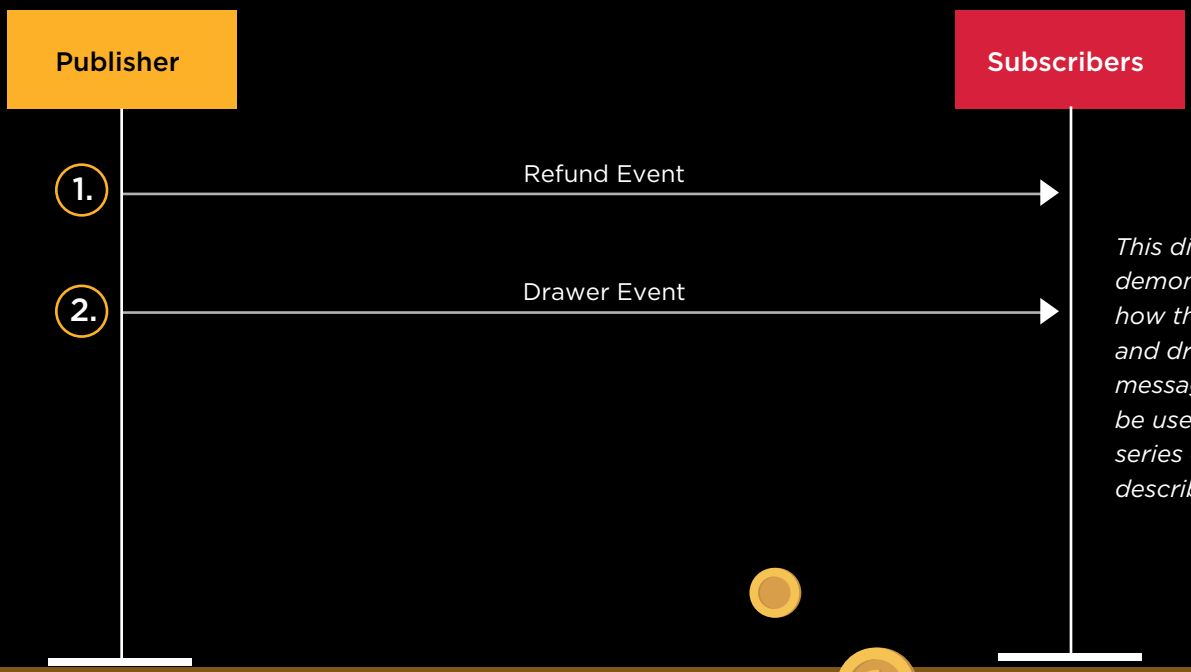
- Menu items exist and are available in the ordering system.
- Employee has clocked in and is available to take the order.
- The till has previously been assigned to the employee, and is ready for transactions.
- Kitchen production, labor management, and inventory management, cash management, and surveillance systems are subscribed to receive these messages.
- The refund is being made on a later business day than when the order was created.

REFUND TRANSACTION (1)

- A refund is created utilizing the refund object.
- The Payment object on the Refund object reflects the amount of the refund as a negative amount.
- The OrderItem object on the order reflects the items being refunded, in this case the pizza and the tax on the pizza reflected as negative amounts.
- Upon completion of the refund, the message is published utilizing the RTN_EventTransactionRQ message with the EventType = Refund.

DRAWER TRANSACTION (2)

- The refund is given to the customer, the RefundID is sent in the ReferencedTransaction field to link the Refund to the drawer transaction (cash refund) and a negative amount is sent in the CashManagement object indicating the removal of cash from the drawer utilizing the RTN_EventTransactionRQ message with the EventType = Drawer.



This diagram demonstrates how the refund and drawer event messages may be used for the series of events as described above.

Use Case 8 Casual Dining/Table Service Order and Payment

- An order is created and a card payment is made at a casual sit-down restaurant (the waitstaff is ringing in each course when they determine they want it produced for the table).
- The waitstaff inputs the order for the two beverages and appetizer into the ordering system, and the order is sent by the ordering system to the restaurant's kitchen to be made.
- Five minutes after the appetizer arrives at the table, the waitstaff adds to the order the two entrees into the ordering system and the order is sent by the ordering system to the restaurant's kitchen to be made.
- After clearing the entrees from the table, the waitstaff enters the order for the dessert into the ordering system and the order is sent by the ordering system to the restaurant's kitchen to be made.
- The customer pays the waitstaff for the order by credit card and the server inputs the payment into the payment system, but keeps the order in an open state allowing for a second transaction to include a tip to be added to the payment.
- Once the tip is added, the waitstaff changes the state of the order to closed.

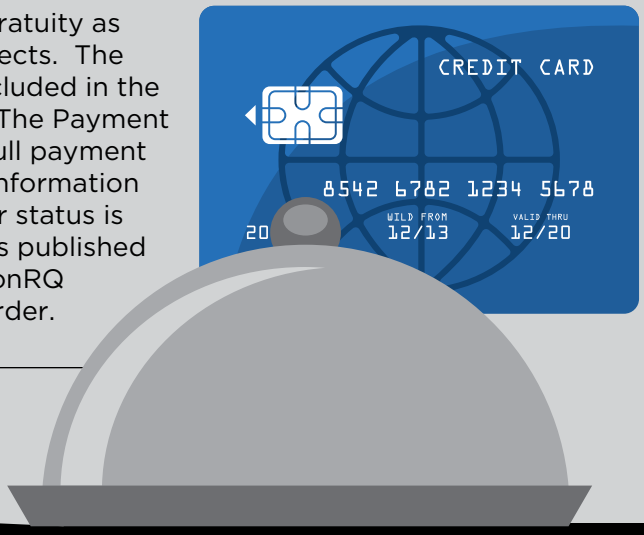
ASSUMPTIONS	• Menu items exist and are available in the ordering system. • Employee has clocked in and is available to take the order. • The till (either physical or virtual) has previously been assigned to the employee and is ready for transactions. • Kitchen production, labor management, and inventory management, cash management, and surveillance systems are subscribed to receive these messages. • Every time the order is committed, the message is cumulative and includes all items from the inception of the order.
ORDER TRANSACTION (1)	• An order is built that includes two glasses of red wine and an order of mozzarella sticks by the ordering system. • The ordering system creates an OrderID and upon completion of the order, the message is published with a status of "sent" each of the items in the order are published with a status of "committed" and received by the kitchen production system utilizing the RTN_EventTransactionRQ message with the EventType = Order.
ORDER TRANSACTION (2)	• The order is updated to include one steak entree and one sea bass entree by the ordering system. • The ordering system includes the OrderID that was created with the initial order and upon completion of the order, the message is published with a status of "sent", each of the items in the order are published with a status of "committed" and received by the kitchen production system utilizing the RTN_EventTransactionRQ message with the EventType = Order.
ORDER TRANSACTION (3)	• The order is updated to include one key lime pie by the ordering system. • The ordering system includes the OrderID that was created with the initial order and upon completion of the order, the message is published with a status of "sent", each of the items in the order are published with a status of "committed" and received by the kitchen production system utilizing the RTN_EventTransactionRQ message with the EventType = Order.

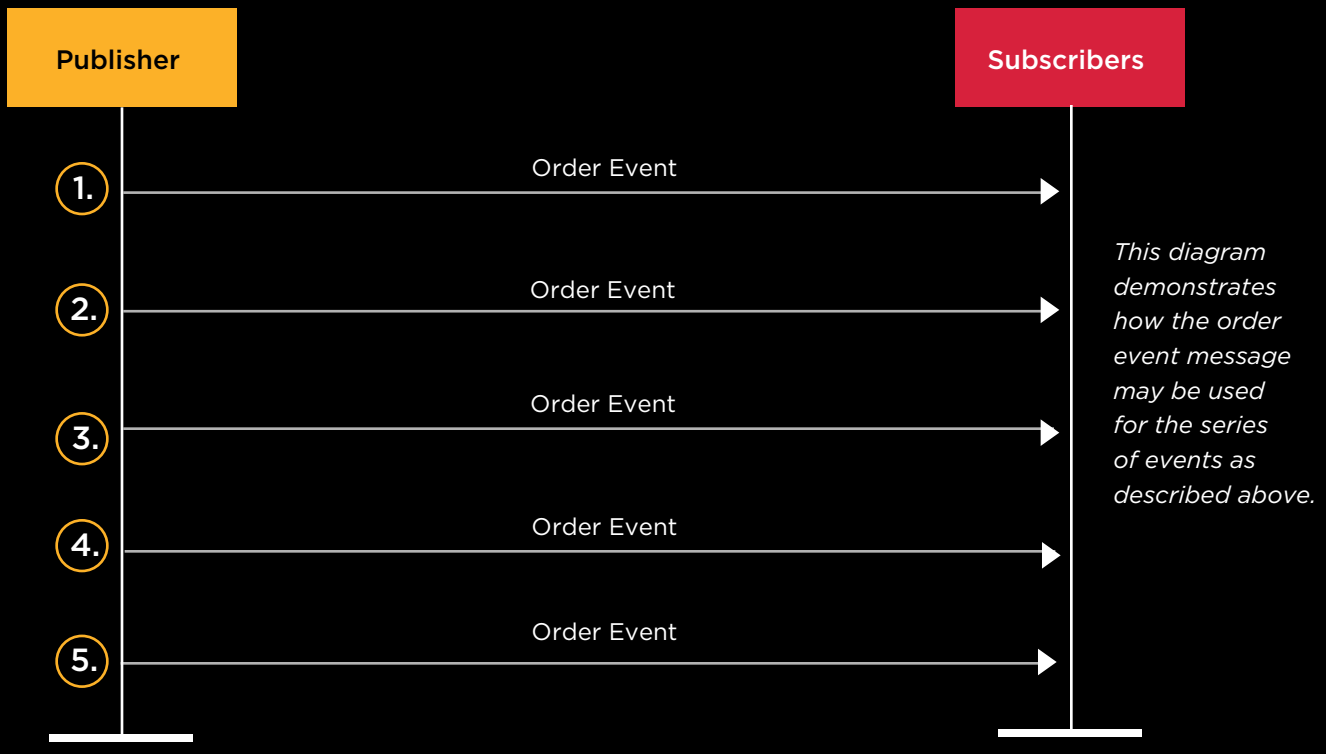
ORDER TRANSACTION (4)

- The order is updated to include payment information.
- The orderID of the original order is included in the OrderID field in the order object. The Payment object on the Order reflects the payment amount and information from the order.
- The order includes each menu item and the cost associated with it and the Total object includes the total cost of all of the menu items. The order status remains "sent" because the server is waiting to complete the order until a gratuity is added to the payment.
- Upon completion of the payment, the message is published utilizing the RTN_EventTransactionRQ message with the EventType = Order.

ORDER TRANSACTION (5)

- The order is updated to include gratuity as part of the payment and total objects. The orderID of the original order is included in the OrderID field in the order object. The Payment object on the Order reflects the full payment amount (including gratuity) and information from the original order. The order status is set to "closed" and the message is published utilizing the RTN_EventTransactionRQ message with the EventType = Order.

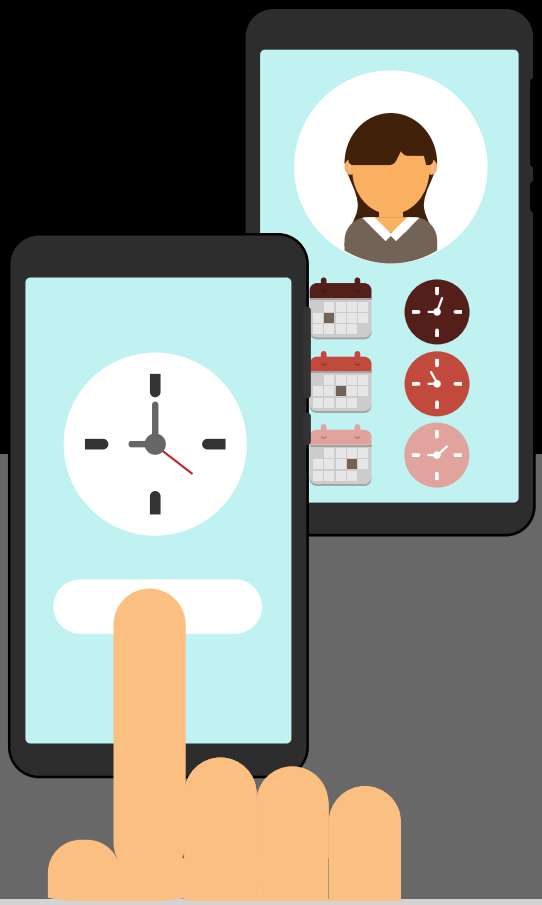




This diagram demonstrates how the order event message may be used for the series of events as described above.

Transactional Data Categories

- Labor
- Scheduling



Use Cases

The remaining pages describe several end-to-end use case scenarios that utilize labor & scheduling, along with diagrams for clarification.

Use Case 1

Publish Work Schedule

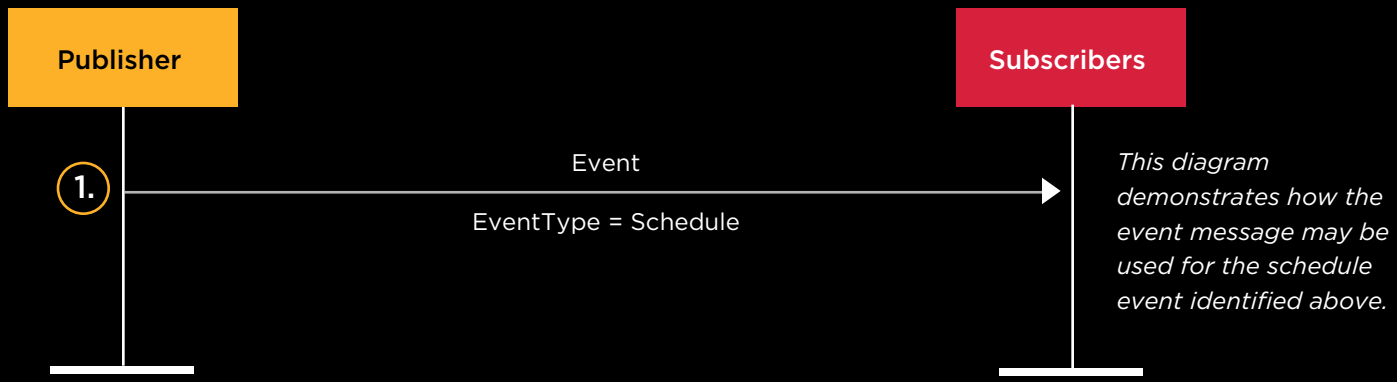
A scheduling system publishes work schedule details, the subscribing time clock system receives the data.

ASSUMPTIONS

- The employees exist in the scheduling system
- The employees are eligible to work
- The timeclock system is subscribed to receive updates from the scheduling system
- The timeclock system maintains a list of employee information, including the employee ID

SCHEDULE DATA EVENT

- The scheduling system publishes a schedule event generating a unique identifier for the event (TransactionGUID) for this event. The schedule is published using the Event message with the EventType of “Schedule” and is received by the subscribing systems.



This diagram demonstrates how the event message may be used for the schedule event identified above.

Use Case 2

Scheduling System Publishes Final Schedule to Reporting System for Reconciliation

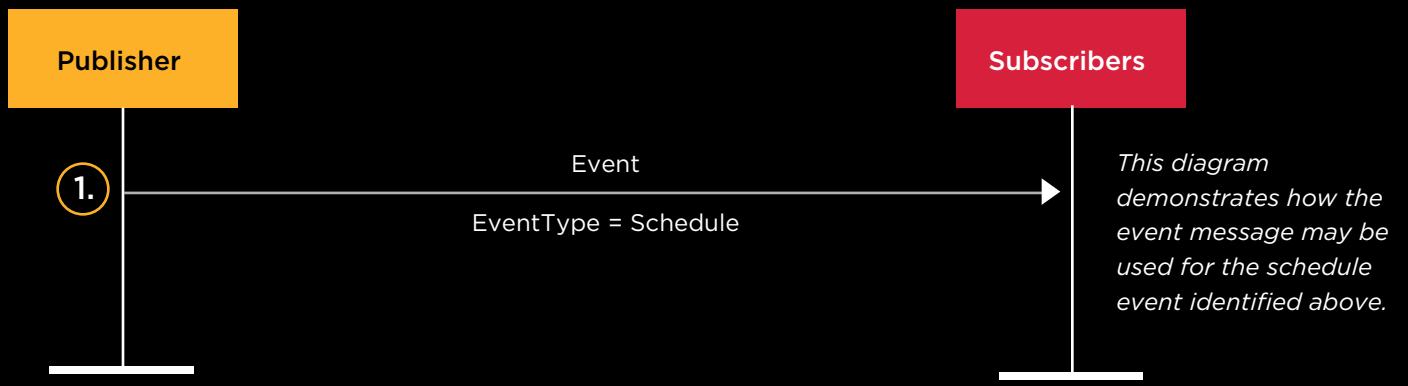
The scheduling system publishes the schedule at the completion of the week to the reporting system to be reconciled with the time clock data.

ASSUMPTIONS

- The employees exist in the scheduling system
- The employees are eligible to work
- The reporting system is subscribed to receive events from the scheduling system
- The reporting system maintains a list of employee information, including the employee ID

SCHEDULE DATA EVENT

- The scheduling system publishes a schedule event, generating a unique identifier for the event (TransactionGUID) for this event. The schedule is published using the Event message with the EventType of "Schedule" and is received by the subscribing systems.



Use Case 3

Time Card System Publishes Time Card Data for the Week to the Reporting System for Reconciliation

The time card system sends a notification to the reporting system of the punch card transactions from the previous week.

ASSUMPTIONS

- The employees exist in the time card system
- The employees are eligible to work
- The reporting system is subscribed to receive messages from the time card system
- The reporting system maintains a list of employee information, including the employee ID

TIME CARD DATA EVENT

- The timecard system publishes the previous week's time cards generating a unique identifier (TransactionGUID) for the event at the completion of the week utilizing the Event message with the EventType of "TimeCardSummary" to be received by the subscribing systems.

Use Case 4

The Human Resources System Publishes Job Codes for Consumption by the Time Card System

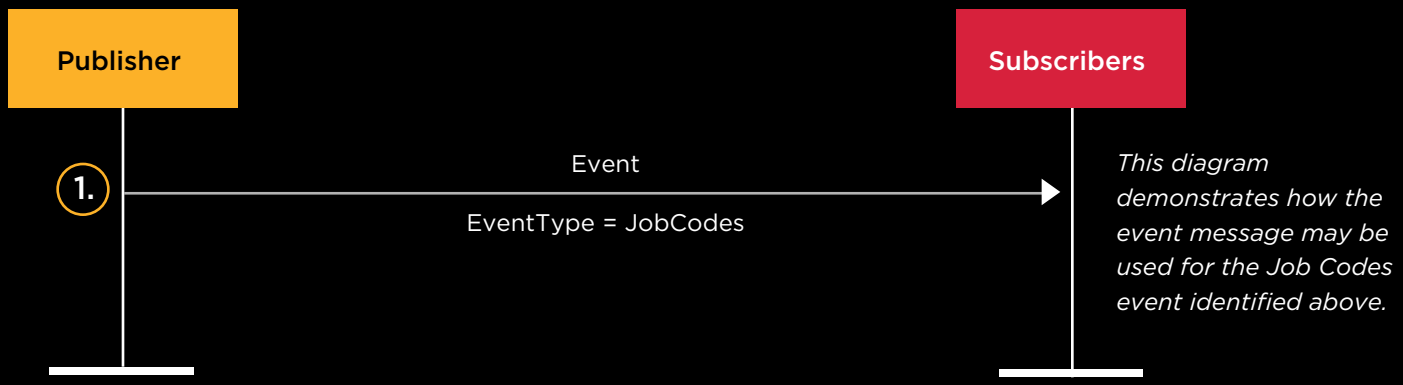
The Human Resources System publishes a list of job codes and their descriptions to subscribers.

ASSUMPTIONS

- Configuration information is understandable by both systems
- The time card system is subscribed to receive messages from the human resource system

PUBLISH JOB CODES

- The human resource system publishes a list of job codes and their corresponding job descriptions generating a unique identifier (TransactionGUID) for the event utilizing the Event message with the EventType of "JobCodes" to be received by the subscribing systems.



This diagram demonstrates how the event message may be used for the Job Codes event identified above.

Use Case 5

Human Resource System Publishes Employee Jobs for Consumption by the Time Card System

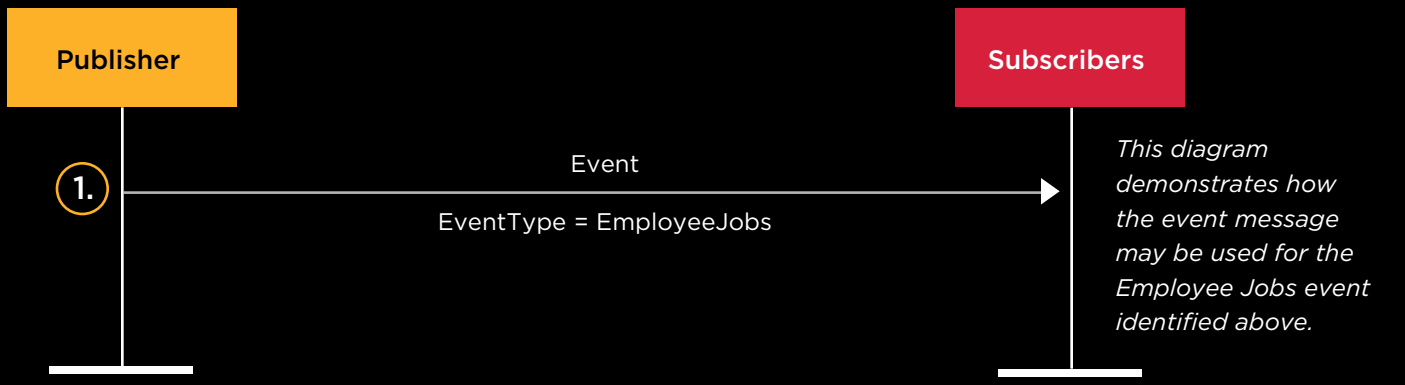
The HR system publishes a list of employee IDs and the eligible jobs for each employee to subscribers.

ASSUMPTIONS

- Configuration information is understandable by both systems
- The time card system is subscribed to receive messages from the human resource system

SCHEDULE DATA EVENT

- The human resource system publishes a list of Employee IDs and the job codes each employee is eligible to work generating a unique identifier (TransactionGUID) for the event utilizing the Event message EventType of "EmployeeJobs" and is received by the subscribing systems.



This diagram demonstrates how the event message may be used for the Employee Jobs event identified above.

Restaurant Technology Network

join. share. thrive.

RTN VISION

In an industry built on service and entrepreneurial spirit, purpose-built technology fuels success. The Restaurant Technology Network aspires to help restaurant professionals and solution providers work together to solve problems large and small and inspire bold ideas for the future.

ADDITIONAL GRAPHICS/ CHARTS RELATED TO THIS STANDARD

+ VIEW ONLY

RELATED RTN RESTAURANT TECHNOLOGY SPECIFICATIONS

Click below to explore.

JOIN US

If you have a vested interest in the restaurant technology industry, join us. Collectively, our members shape the industry by creating and disseminating technology standards and technical guidance to benefit members. Through our cornerstone virtual think-tank workgroup meetings, our members solve industry challenges and prosper inside a unique, collaborative environment.

+ VIEW OUR MEMBERS

WANT FIRST-RIGHTS ACCESS TO DOCUMENTS LIKE THIS? JOIN RTN TODAY.

+ CHECK OUT NEW SITE

www.restauranttechnologynetwork.com